Remarks

Applicants respectfully request reconsideration of the present Application. Claims 1, 9, 11, 16, 25, 26, and 28-33 have been amended herein. Further, claims 10, 15, and 27 have been canceled. Care has been exercised to introduce no new matter. Claims 1-9, 11-14, 16-26, and 28-33 are pending and are in condition for allowance.

Objections based on 35 U.S.C. § 101

In the Office Action mailed September 3, 2008, claims 1, 9, and 33 were objected to as not being fully directed to statutory subject matter. In response, claims 1, 9, and 33, have been amended to recite "computer readable storage medium...." Similarly, claim 25 has also been amended to recite "computer readable storage medium...." Applicants submit that these amendments overcome the claim objections. As such, Applicants respectfully request withdrawal of these objections.

Response to Arguments

In the Office Action, it was noted that the continuation-in-part priority date (February 28, 2003) was not acknowledged because Applicants had not established common ownership. Office Action, p. 4. In response, Applicants note that common ownership is unnecessary to establish a right of priority. *See* MPEP 201.08. Applicants submit that it is entitled to the continuation-in-part priority date of February 28, 2003, and such favorable action is requested.

Rejections based on 35 U.S.C. § 103(a)

A) **Applicable Authority**

To sustain a rejection of a claim under 35 U.S.C. § 103(a), the Examiner must find that a preponderance of the evidence supports a finding of obviousness. The Examiner

bears the initial burden of showing that the reference teachings establish a prima facie case of obviousness. "In view of all factual information, the examiner must . . . make a determination whether the claimed invention 'as a whole' would have been obvious at that time to that person." MPEP § 2142 (explaining that "[t]he key to supporting any rejection under 35 U.S.C. 103 is the clear articulation of the reason(s) why the claimed invention would have been obvious"). In making that determination, the Examiner must consider every word in each claim. *In re Wilson*, 424 F.2d 1382, 1385, 165 USPQ 494, 496 (CCPA 1970).

Recently, the Supreme Court indicated that "it will be necessary for [the Office] to look at interrelated teachings of multiple [prior art references]; the effects of demands known to the design community or present in the marketplace; and the background knowledge possessed by [one of] ordinary skill in the art . . . to determine whether there was an apparent reason to combine the known elements in the fashion claimed by the [patent application]." *KSR v. Teleflex*, 127 S. Ct. 1727 (2007). However, if the references do not "expressly or impliedly suggest the claimed invention," the examiner must present "a convincing line of reasoning as to why the artisan would have found the claimed invention to have been obvious in light of the teachings of the references." MPEP § 706.02(j) and § 2142, quoting *Ex parte Clapp*, 227 USPQ 972, 972 (Bd. Pat. App. & Inter. 1985).

B) <u>The rejection of claims 1-33 under 35 U.S.C. § 103(a) as being obvious over Walker, U.S. Publication No. 2004/0015840 in view of Bhansali *et al.*, U.S. Publication No. 2002/0169999 should be withdrawn because the references cannot be combined</u>

In the Office Action, claims 1-33 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Walker, U.S. Publication No. 2004/0015840 (hereinafter "Walker") in view of Bhansali *et al.*, U.S. Publication No. 2002/0169999 (hereinafter "Bhansali"). Applicants submit

that proposed combination of references cannot be combined to support a prima facie case of obviousness. As such, Applicants respectfully traverse this rejection, as hereinafter set forth.

Independent claim 1 is generally directed to a compiling system embodied on a computer readable storage medium for compiling a markup language file into an executable application. The compiling system includes a parser for parsing the markup language file and providing the compiling system with detailed token information. The compiling system further includes a code generator for generating a language-independent tree of code expressions based on the token information. The code expressions represent the markup file as a class. The compiling system still further includes **a compiler** for compiling the code expressions to create the executable application.

Contrary to the invention of independent claim 1, the Walker reference discloses a method and mechanism for converting between JAVA classes and XML. *See* Walker, Abstract. In particular, the method disclosed in the Walker reference allows programmers to "convert easily between JAVA and XML representations of data while working exclusively with classes from the application domain. The invention enables developers to easily add XML support for complex hierarchies of any user-defined class, JAVA primitives (int, float, boolean, etc.) and wrapper classes (Integer, Boolean, Float, etc.) as well as collections and arrays of such objects." *Id.*, at ¶ [0018]. As a result of this easy conversion, the invention disclosed in the Walker reference "**avoids the use of a schema compiler**" to generate new JAVA classes. *Id.*, at ¶ [0019]. As this passage makes clear, the Walker reference expressly teaches away from using a schema compiler, describing them as "problematic" and "limiting." *See also id.*, at ¶ [0003].

The Bhansali reference discloses a compiler that completes compilation upon identifying an unresolvable input during compilation instead of aborting compilation. Bhansali,

Abstract. The invention of Bhansali achieves this by incorporating an exception-throwing instruction(s) in the output code of the compiler. *Id.*, at ¶ [0020].

Based on at least the foregoing, Applicants submit that the combination of the Walker and Bhansali references is improper as these references teach away from such a combination. "It is improper to combine references where the references teach away from their combination. *In re Grasselli*, 713 F.2d 731, 743, 218 USPQ 769, 779 (Fed. Cir. 1983)." MPEP § 2145. The Bhansali reference is concerned with correcting and/or avoiding errors made during the compilation process. Of course, correcting and/or avoiding errors made during the compilation process necessarily requires a compiler compiling code. This directly conflicts with the approach in the Walker reference, in which "problematic" and "limiting" compilers are expressly "avoid[ed]." *See, e.g.,* Walker, at ¶ [0019]. Common sense alone says that methods for avoiding a "problematic" compiler are not particularly relevant to an approach that intentionally incorporates such compilers. There is clearly no suggestion or motivation to combine two such differing systems. Accordingly, for at least the above-noted reasons, the combination of references is improper and should be withdrawn.

Applicants also submit that there is no suggestion or motivation to modify the Walker reference with the Bhansali reference because the modification would render the invention in the Walker reference unsatisfactory for its intended purpose. "If [a] proposed modification would render the prior art invention being modified unsatisfactory for its intended purpose, then there is no suggestion or motivation to make the proposed modification. *In re Gordon*, 733 F.2d 900, 221 USPQ 1125 (Fed. Cir. 1984)." MPEP § 2143.01. The intended purpose of the system in the Walker reference is to convert XML into JAVA (or vice versa) **without using a "problematic" or "limiting" compiler**. *See, e.g.,* Walker, at ¶¶ [0003] and

[0019]. To modify the Walker reference with the Bhansali reference or any reference disclosing a compiler would simply render Walker unsatisfactory for its intended purpose. For example, the Walker reference states that "the use of a schema compiler is problematic and limiting to a developer in terms of control and flexibility of the mapping of JAVA classes to XML." *Id.*, at ¶ [0003]. The Office Action attempts to modify the Walker reference by incorporating the teachings of the Bhansali reference with the Walker reference. As previously indicated, the Bhansali reference "relates to compilers, and more particularly, the handling of compilation exceptions." Bhansali, at ¶ [0001]. Accordingly, the incorporation of the disclosure from the Bhansali reference into the Walker reference suggests using a compiler to compile code in the XML to JAVA conversion. However, the system in the Walker reference intentionally excludes using a "problematic" compiler during the conversion process. As such, Applicants respectfully submit that the modification would render the system in the Walker reference unsatisfactory for its intended purpose, and thus there is no suggestion or motivation to modify the Walker reference by incorporating teachings from the Bhansali reference.

Additionally, Applicants submit that there would be no suggestion or motivation to modify or combine the Walker reference with the Bhansali reference because it would change the principle of operation of the system in the Walker reference. "If the proposed modification or combination of the prior art would change the principle of operation of the prior art invention being modified, then the teachings of the references are not sufficient to render the claims *prima facie* obvious. *In re Ratti*, 270 F.2d 810, 123 USPQ 349 (CCPA 1959)." MPEP § 2143.01. The principle of operation of the system in the Walker reference is to convert XML into JAVA or JAVA into XML **without using a compiler**. To modify the Walker reference with the Bhansali reference or any reference disclosing a compiler would simply destroy the principle of operation

of the Walker system because such a combination would require incorporating a compiler into a system intended to operate without a schema compiler; thereby introducing the "problem[]" the Walker reference was attempting to avoid. Accordingly, Applicants respectfully submit that the modification would change the principle of operation of the system in the Walker reference, and thus there is no suggestion or motivation to modify or combine the Walker reference with the Bhansali reference or any reference disclosing a compiler.

Based on the disclosure of Walker, Applicants further submit that the Walker reference cannot serve as a proper reference against the claimed compiling systems and/or compiling methods. To modify the Walker reference in such a manner would violate the express teaching of the Walker reference, would render the invention in the Walker reference unsatisfactory for its intended purpose, and would modify the principle of operation disclosed by the Walker reference.

As the Office Action fails to state a prima facie case of obviousness of claim 1, for at least the reasons described hereinabove, Applicants respectfully request that the rejection of this claim be withdrawn and claim 1 be allowed.

Independent claim 9, as amended herein, is generally directed to a compiling system embodied on a computer storage readable medium for compiling a markup language file into an executable application. The compiling system includes a parser for parsing the markup language file and providing the compiling system with detailed token information including non-code token information to the compiling system. The markup language file is associated with at least one code-behind file. The compiling system further includes a binary file generator for generating a binary file from non-code token information. The binary file contains one record for each non-code token. The compiling system also includes a code generator for generating a

language-independent code expression that represents the markup language file as a class. The compiling system still further includes an application generator for compiling the code files into an executable application.

Independent claim 17 is generally directed to a method for compiling a markup language file into an executable application. The method includes receiving a markup language file. The method further includes parsing the markup language file and providing a compiling system with detailed token information. The method also includes generating a language-independent tree of code expressions based on the token information. The code expressions represent the markup language file as a class. The method still further includes compiling the code expressions to create the executable application.

Independent claim 26, as amended herein, is generally directed to a method for compiling a markup language file into an executable application. The method includes parsing the markup language file and providing the compiling system with detailed token information including non-code token information. The method also includes generating a binary file from the non-code token information. The binary file contains one record for each non-code token. The method also includes generating a language-independent code expression that represents the markup language file as a class. The method still further includes compiling the code expressions into an executable application.

For at least the reasons identified with respect to independent claim 1, Applicants respectfully submit that the proposed combination fails to state a prima facie case of obviousness of claims 9, 17, and 26, for at least the reasons described hereinabove. As such, Applicants respectfully request that the rejections of these claims be withdrawn and the claims be allowed.

Claims 2-8, 11-14, 16, 18-25, and 28-33 depend, either directly or indirectly, from one of allowable independent claims 1, 9, 17, and 26. Accordingly, it is respectfully submitted that the Walker and Bhansali references fail to render these claims obvious for at least the reasons stated above. As such, a withdrawal of the § 103(a) rejections of claims 2-8, 11-14, 16, 18-25, and 28-33 is respectfully requested. Each of claims 2-8, 11-14, 16, 18-25, and 28-33 is believed to be in condition for allowance, and such favorable action is respectfully requested.

C) **The rejection of claims 1-33 under 35 U.S.C. § 103(a) as being obvious over Walker in view of Bhansali should be withdrawn because the proposed combination fails to disclose each and every limitation of amended claim 9**

Independent claim 9, as amended herein, is generally directed to a compiling system embodied on a computer storage readable medium for compiling a markup language file into an executable application. The compiling system includes a parser for parsing the markup language file and providing the compiling system with detailed token information including non-code token information to the compiling system. The markup language file is associated with at least one code-behind file. The compiling system further includes a binary file generator for generating a binary file from non-code token information. The binary file contains one record for each non-code token. The compiling system also includes a code generator for generating a language-independent code expression that represents the markup language file as a class. The compiling system still further includes an application generator for compiling the code files into an executable application.

In contrast, as described above, the Walker reference discloses a method and mechanism for converting between JAVA classes and XML. *See* Walker, Abstract. In particular, the method disclosed in the Walker reference allows programmers to "convert easily between JAVA and XML representations of data while working exclusively with classes from

the application domain. The invention enables developers to easily add XML support for complex hierarchies of any user-defined class, JAVA primitives (int, float, boolean, etc.) and wrapper classes (Integer, Boolean, Float, etc.) as well as collections and arrays of such objects." *Id.*, at ¶ [0018]. However, the Walker reference utterly fails to disclose at least the limitation that the "markup language file is associated with at least one code-behind file." On the contrary, the Walker reference discloses converting XML code into JAVA class representations. The conversion process of Walker fails to disclose associating a markup language file with one or more code-behind files.

Likewise, the Bhansali reference fails to teach or suggest each limitation of independent claim 9. As stated above, the Bhansali reference discloses a compiler that completes compilation upon identifying an unresolvable input instead of aborting compilation. Bhansali, Abstract. The invention of Bhansali achieves this by incorporating an exception-throwing instruction(s) in the output code of the compiler. *Id.*, at ¶ [0020]. However, the Bhansali reference simply fails to discuss markup languages; let alone associating a markup language file with at least one code-behind file. Applicants note that the Office Action recognized that Bhansali failed to disclose markup languages, as the Office Action did not cite Bhansali for this purpose. As such, Applicants submit that Bhansali fails to cure the deficiencies of the Walker reference.

In view of the foregoing, it is submitted that the Walker reference in view of the Bhansali reference fails to teach or suggest all the limitations of independent claim 9, as herein amended. Accordingly, Applicants respectfully request withdrawal of the § 103(a) rejection regarding independent claim 9.

Further, Applicants note that each of claims 11-14 and 16 depend directly from allowable independent claim 9. Therefore, claims 11-14 and 16 are patentable for at least the reasons identified above with reference to claim 9. Moreover, each of claims 11-14 and 16 further recite other novel and nonobvious features neither taught nor suggested in the cited references in the context of these claims. Accordingly, Applicants respectfully request withdrawal of all § 103(a) rejections regarding claims 11-14 and 16. Claims 11-14 and 16 are believed to be in condition for allowance and such favorable action is respectfully requested.

CONCLUSION

For at least the reasons stated above, claims 1-9, 11-14, 16-26, and 28-33 are now in condition for allowance. Applicants respectfully request withdrawal of the pending rejections and allowance of the claims. If any issues remain that would prevent issuance of this application, the Examiner is urged to contact the undersigned – 816-474-6550 or ahankel@shb.com (such communication via email is herein expressly granted) – to resolve the same. It is believed that no fee is due, however, the Commissioner is hereby authorized to charge any amount required to Deposit Account No. 19-2112.

Respectfully submitted,

/s/ Aaron E. Hankel /

Aaron E. Hankel
Reg. No. 60,663

JSG/AEHX/bp
SHOOK, HARDY & BACON L.L.P.
2555 Grand Blvd.
Kansas City, MO 64108-2613
816-474-6550